ISSN(Online): 2395-7638

International Journal of Recent Research in Science, Engineering and Technology

Vol. 1, Issue 8, November 2015

Load balancing of Data Items in the Clouds

V.R Elaimathi[1], S. Nandagopal[2]

Assistant Professor, Department of CSE, RR Institute of Engineering, Sivaganga, Tamil Nadu,India[1]

Assistant Professor, Department of CSE, RR Institute of Engineering, Sivaganga, Tamil Nadu,India[2]

ABSTRACT: In this paper, we study this data staging problem by leveraging the dynamic programming (DP) techniques to optimally migrate, replicate, and cache the shared data items in cloud systems with or without some practical resource constrains in an efficient way whiling minimizing the monetary cost for transmitting and caching the data items. Monetary cost is our primary interest as on one hand, the provision of the resources in cloud systems are usually based on pay-as-you-go fashion, and thus effective use of the platforms within budget constraint is always the user's concern.

KEYWORDS: Dynamic Programming, Resource Constraints.

I. INTRODUCTION

With increasing data accessibility demands on clouds, data availability maximization seems to be important problem to consider to maintain high-fidelity and time-bounded service expectations in clouds. For example, one of the pressing needs by the cloud service providers (CSPs) is to efficiently serve the needs of the user requests that demand single or multiple data items in the shortest possible time. Thus, with growing population of cloud users, the problem of making the requested data available to the users becomes an imperative issue for CSPs to guarantee high-quality services. An particularly appeal-ing approach to maximizing such data availability is to stage the requested data to some vantage sites and cache the data for a period of time so that the quality of service (e.g., latency minimization, network traffic reduction) for user's future accesses can be greatly improved. We refer to this integrated functionality as data staging. An exemplary scenario is a mobile user who may want a service to suggest probable alternative routes to her destination depending on current traffic patterns. This service requires not only making the queried data highly available at vantage nodes along her current path but also staging the data at key sites on probable paths based on her navigation history. Clearly from this example, data staging could effectively facilitate the service accesses. However, achieving the benefits is not free due to the costs for access pattern detection and data staging. In this paper, we study this data staging problem by leveraging the dynamic programming (DP) techniques to optimally migrate, replicate, and cache the shared data items in cloud systems with or without some practical resource constraints in an efficient way while minimizing the monetary cost for transmitting and caching the data items (glory details are discussed in the later of this section). Monetary cost is our primary interest as on one hand, it is a very flexible concept to reflect the qualities of various network features such as network bandwidth, link latency, and storage utilization, and on the other hand, the provision of the resources in cloud systems are usually based on pay-as-you-go fashion, and thus effective use of the platforms within budget constraint is always the user's concern. Due to the optimality, our solutions are unique and advantageous over other (suboptimal) methods to provide the cloud-based services with the flexibility that they can not only decide the duration of each data item to be cached at some vantage sites but also make a tradeoff between transmission cost and caching cost to meet the constraints imposed by the underlying Infrastructure as a Service Providers (IaaSPs), information item owners or CSPs' budget. As cloud computing is gaining its prominence, we believe these benefits are more important than ever before to the success of the traditional network-based services migrating to clouds such as the distributed collaborative document editing process and multimedia personalized services where a document or video clip may be requested by users in a sequence of predefined time instants.

II. WORK FORMULATION.

Suppose there are k distinct shared data items initially stored at one node, say p1 and later migrated, replicated, and cached in a fully connected network with m nodes (p1,p2,…,pm) to serve a sequence of requests σ Δ= σ1 σ2… σn in which each which each σi Δ= (ti , pi ,Ri), $1 \le i \le n$ is specified by the predicted access pattern and represents a request made for a data subset Ri by node pi at time ti . We therefore have the complete knowledge of all such information as an input to our algorithms. Further, for simplicity we also assume that there exists only one request per stage. To satisfy a request for a particular data item, we define the following primitive operations to preform on the cached data items, which may involve caching and transmission costs:

1.retainment: cache the data item at a node pu from time tu to tv by paying (tv _ tu)Su, Su is the rate of caching cost at node pu, $1 \le u \le m$. 2. migration: move the data item from a node pu to a node pv at a cost equal to the distance Cuv. 3. replication: copy the item to the request node pv from a node pu at a cost of Cuv. 4. excursion: satisfy the request at a node pv by using the copy at a node pu without migration at a cost of Euv. 5. creation/deletion: create/delete the selected copies at some nodes without incurring any cost. We thus follow this principle in our model to allow Euv = Cuv for any pair of pu and pv in the network. Each new request can trigger one or more operations, leading to the creation and deletion of the data items at arbitrary nodes.

III. OPTIMAL DATA STAGING ALGORITHMS

1. **Multiple Copies without Constraints**

Our multicopy algorithm for handling k distinct data items with total cost is shown in DP Recrusion (2) where Fj(i) is maximum total cost for accessing item j till $t_i \ge 1$:

Recursion indicates that when there is no request for item j made t_i (j € R $_i$)the minimum cost till t_i depends on the source point set whose elements are candidate copies to service the current request, and the shortest path from the selected source point to the current request point. In addition, we also need to update the costs for the request points in the affected set because of the side effects of the shortest path computation. The final scheme for the data staging obtained from Recursion contains all the information pertain to how primitive operations are combined to achieve the minimum cost.

2. **Resource Constraints on Per-Item Basis**

Since the resource concepts of transmission and copy in our research are highly related, in this section we uniformly investigate the optimality when the constraints are given on per-item basis to serve the entire request sequence (i.e., on per-scheduling-cycle basis).

3. **Dynamic Programming Recursion**

For description, we define Fj(i, cj i) as the minimum cost for item j till stage ti given the maximum number of resource c j i (cj 0= c j), here the resource is either the number of copies or the number of transmissions. The total cost for all k items at ti is F(i)=$\sum_{i=1}^{k}$ Fj(i,cj i) where Fj(I,ci j) defined by DP . The recursion is not difficult to understand. If no request for item j is made at ti , the total cost remains unchanged, otherwise, the resource at stage ti will be deducted by x units from the current available c j i and the remaining is applied to some source point l.

The x units are then distributed among computing the shortest path cost (i.e., y units) and the costs for all elements in the affected point set (i.e., z units). The minimum sum of these three parts over the source point set Ei j and the current available resources c j i is the total cost until stage ti . As we defined, the source point set is a constant set, the minimum cost hence can be obtained independent of the scheduling path.

Note that SP (u, v, c) returns + ∞ if the resource c is not sufficient enough to compute any constrained shortest path. In this situation, Fj(i,cj) is also failed to compute and returns + ∞ as well. The optimal solution is Fj(n, cj). Obviously, when we have the transmission constraints, more copies will be used. Similarly, when we have a constraint on the copies, more transmissions will be used.

ISSN(Online): 2395-7638

International Journal of Recent Research in Science, Engineering and Technology

Vol. 1, Issue 8, November 2015

IV. ON HETEROGENEOUS COST MODEL

Up to now we have investigated the data staging problem under the homogeneous cost model, together with some practical constraints. For completeness, in this section we briefly discuss the problem under the heterogeneous cost model where each node has its own caching cost rate, and the transmission cost rate between any pair of nodes is not always identical. As this general case is overly complex and somehow connected to the Rectilinear Steiner Arborescence (RSA) problem it is generally believed to be an NP-Complete problem .However, by extending the DP algorithm where Sk ≥Cij, i, j, k = 1, ..m is considered for optimal data staging solution, we can show that this heterogeneous case is still tractable under some restricted conditions. In the following discussion, we concentrate on a single data item thereby designing our algorithms to this problem.

1. Configuration Distance

We define a metric space (M,d), which is the space-time diagram with a distance d(u,v) defined as the shortest path between any pair of points (pu,tu) and (pv ,tv) in the space-time diagram. A k-configuration for (M,d) is any multiset C of size k over M. In this section, we focus on the computation of the minimum cost of transforming one _-configuration to another.

2. Definitions

Suppose Ci and Cj are two k-configurations then d(Ci ,Cj) is the distance between Ci and Cj , which is the minimum distance/cost traveled/spent by all the data items and their copies that change configuration from Ci to Cj . where π is the permutation of {1,2,…k}. In general, the computation of d(pl i , pπ (l)) in (M,d) is very complicated as it needs to map a set of nodes at t-1 (i.e., Ci) to another set of nodes at t (i.e., Cj) given that the mapping paths may interference with each other for a minimum cost. Fig. 2a is an example to illustrates the difficulties in computing d(Ci ,Cj) where Ci ={ a,b,c } will map to Cj={1,2,3} with a minimum cost. In this example, b first mapping to 2 and then making a replication at 3 are cheaper than mapping b; c to 2 and 3, respectively. Apparently from this example, figuring out the optimal mapping paths between a pair of configurations is the key point to attain the distance.

3. Bijective Mapping Function

There are two major issues in the above network-flow algorithm. One is the restriction on the cost model and the other is to allow multiple continuous replications to shift Ci to Cj at time t (see the reason why copy c in the example Fig. 2a is not involved in the mapping). Although the restricted cost model is not always satisfactory, it is still acceptable under certain conditions. In contrast, making multiple replications at a time instant is in general infeasible in practice as the resource cost of replication is usually high. Therefore, in the following discussion we always assume that the function that maps Ci to Cj from t -1 to t is a bijective function, which technically prohibits the multiple replications from happening simultaneously.

4. Greedy Algorithm

Not surprisingly, due to the hardness of this problem, the time complexity of the DP-based algorithm is prohibitively high even though we have imposed some constraints on the problem. In this section, we propose a simple greedy algorithm to the problem in its general form, whose staging cost is at most twice the optimum. The basic idea is very simple. Given σ=σ1,σ2..σn, we first construct a service graph G(V,E) in the following way: 1. V = {1,2. . . . n} represents the n requests making at (pi, ti), i € [1, . . . , n] 2. for each request i, compute its source point set Ei 3. for each point l € Ei, add an arc from l to i in G with the weight being equal to the cost of the shortest path from l to i We compute a minimum spanning branching (MSB) of the service graph to satisfy all request demands. Since |Ei |≤ m, the total number of arcs is not greater than O(mn). and thus the time complexity of constructing the service graph is O(mn).

V. EMPIRICAL STUDIES

ISSN(Online): 2395-7638

International Journal of Recent Research in Science, Engineering and Technology

Vol. 1, Issue 8, November 2015

The solver can be configured by several parameters including the size of network, the number of distinct data items, and the available resources. It also accepts as an input a sequence of request demands that is made for any subset of k distinct data items. Of course, there constantly exist studies on modeling the generation of various access sequences in different research contexts and scenarios .However, this kind of information is irrelevant to our empirical studies as the sequence of request has been presumably known in advance. Unlike the sequence of requests, there is no such a readily available model that is well accepted to specify the distribution of the accessed items in each request. To address this issue, we intentionally allow that the requests for each item are automatically generated by following some probability distribution with item-dependent parameters to mimic the real case where the access opportunities among the k items are not equal. More specifically, we assume that the request for item j at a particular time step follows the Bernoulli distribution with a successful probability.

1. Impact of Caching and Transmission Rates on Total Costs

We first investigate the impact of caching and transmission rates on the total costs. To this end, we fix a 50-node fully connected network, and conduct a balanced and a unbalanced request sequences of length 100 by following the discussed methods. Each request is made for at most 10 data items. Fig. 4 shows our experimental results when the maximum number of copies is restricted by 10 and transmission constraint of 308, respectively. These numbers allow the requests to have the minimal resources to access the required data items. Both caching and transmission rates in all cases have linear impact on the final costs. These observations are easy to understand since the total cost undergoes a proportional increase with the number of copies and transmissions.

Comparing with balanced and unbalanced requests, balanced requests under both constraints result in more costs due to their wider dispersion among the network nodes than their unbalanced counterpart. In addition, another interesting observation is the comparison between the two constraints. Although the two kinds of resources are minimum to both balanced and unbalanced requests, the transmission constraints will lead to more total costs than the copy constraints, especially for the unbalanced requests. This phenomenon is not as easy as that of the compared request distribution to understand because the constraints on one type of resources will naturally increase the use of the other type of resources to serve the entire request sequence, and both types of resources have the same range of cost rates. However, we found that the rationale behind this phenomenon is the time-invariant transmission cost to serve a request, which is different from the time-dependent caching cost. As a consequence, when transmissions are restricted, the algorithm needs to keep more cached copies for the subsequent accesses, which, by and large, increases the total costs compared with limiting the number of copies where each used transmission only has a constant cost rate C. Such benefits of copy constraints are more pronounced for the unbalanced requests due to the relative bad performance of transmission constraints where some nodes may accept a quite few requests, and thus the first copy maybe needed to kept for a long time for accesses.

2. Impact of Copy Constraints on Total Costs and Solver Performance

By following the same configuration in the last experiment, in this section, we conduct experiments to study the impact of available resources on total costs and evaluate the performance of the solver by measuring its execution time. Given constraints on the number of copies, Table 2 shows our results for both balanced and unbalanced requests when fixing C = 60 and varying S from 5 to 120. The given numbers of copies are optimally distributed among the 20 requested data items to minimize the total cost. Comparing the balanced and unbalanced cases, we found that the reduction is relatively significant for unbalanced requests. We think this is because if more copies are allowed, for most of the unbalanced requests, the algorithm is easier than in the balanced case to find a nearby copy with a minimum cost to serve a request. This explanation can be verified by observing the situations when C=S is high in which the cached copies are usually biased toward serving requests. This also explains why as C=S decreases, the benefits of adding more resources are gradually diminished. although the total costs decrease linearly as the copy numbers increase, the staging time increases in a nonlinearly quadratic fashion, which verifies our analysis on the time complexity where given the fixed number of nodes, the length of request sequence as well as the total number of distinct data items, the time complexity of our algorithm is a quadratic polynomial in the number of copies.

3. Empirical Studies on Transmission Constraints

ISSN(Online): 2395-7638

International Journal of Recent Research in Science, Engineering and Technology

Vol. 1, Issue 8, November 2015

In addition to the copy constraints, we also conducted the same set of empirical studies on the transmission constraints. Like what we have observed the results under the transmission constraints more or less exhibit the same relation forms as those under the copy constraints, which are consistent with our intuition as the copy and transmission are highly inter-related for data staging.

VI. CONCLUSIONS

In this paper, we studied the problem of staging a set of distinct data items in a fully connected network to facilitate cloud-based services with minimum cost. To this end, we investigated the optimal staging strategies based on the cost models in the paper to minimize the total staging cost. We also considered some practical constraints on this problem in terms of the maximum number of transmissions and copies. We achieved an efficient optimal solution via dynamic programming to the situation when the numbers of Transmissions and copies are unbounded within the time complexity of O(kmn2). When S > C, this algorithm also ensures an optimal single-copy algorithm within the complexity of O (kn2). For arbitrary S and C, our results show that when C=S is low, a single copy of each data item can efficiently serve all the user request sequence. In addition to the homogeneous cost model, we also briefly discuss this optimization problem under a heterogeneous cost model, which is generally believed to be NP-complete. We proposed a DP-based algorithm which is efficient under some restrictions and presented a 2-approximation algorithm to the general case. In this paper, we assumed that the accesses to the k distinct items are independent with each other and the cost model is defined on per-item basis. Although these assumptions can simplify the algorithm design, it is not always sufficient to reflect the reality where some association rules may exist among the user accesses and caching or transmitting the data items in bulk could be cheaper than in each item individually. We will study these issues issues in the future work. Considering this problem under the storage constraints is another interesting problem. As each node is equipped with fixed storage size, the caching cost would be time dependent and not increase linearly with caching time. Therefore, solutions to the problem with this constraints is more practical in reality and thus worthwhile to study in the future.

REFERENCES

[1] G. Pei and M. Gerla, “Mobility management for hierarchical wireless networks,” Mob. Netw. Appl., vol. 6, no. 4, pp. 331–337, 2001.
[2] T. Ohta, M. Fujimoto, S. Inoue, and Y. Kakuda, “Hi-tora: a hierarchical routing protocol in ad hoc networks,” in p. 143, 7th IEEE International Symposium on High Assurance Systems Engineering (HASE), 2002.
[3] B. Karp and H. T. Kung, “Gpsr: greedy perimeter stateless routing for wireless networks,” in MobiCom ’00: Proceedings of the 6th annual international conference on Mobile computing and networking. New York, NY, USA: ACM Press, 2000, pp. 243–254.
[4] J. Eriksson, M. Faloutsos, and S. Krishnamurthy, “Scalable ad hoc routing: The case for dynamic addressing,” in proceedings of IEEE INFOCOM, 2004.
[5] G. Pei, M. Gerla, and X. Hong, “Lanmar: landmark routing for large scale wireless ad hoc networks with group mobility,” in MobiHoc ’00: Proceedings of the 1st ACM international symposium on Mobile ad hoc networking & computing. IEEE Press, 2000, pp. 11–18.
[6] Y. Ge, L. Lamont, and L. Villasenor, “Improving scalability of heteroge- neous wireless networks with hierarchical olsr,” in The OLSR Interop & Workshop, 2004.
[7] Z. J. Hass, “A new routing protocol for the reconfigurable wireless net- works,” in Proceedings, IEEE 6th International Conference on Universal Personal Communications, 1997, pp. 562–566.
[8] N. Nikaein, C. Bonnet, and N. Nikaein, “Harp - hybrid ad hoc routing protocol,” in proceeding of IST ’01: International Symposium on Telecom- munications, 2001.